BINARY STATE MACHINE SYSTEM AND METHOD FOR REGEX PROCESSING OF A DATA STREAM IN AN INTRUSION DETECTION SYSTEM

TECHNICAL FIELD OF THE INVENTION

The present invention relates in general to computer network intrusion detection systems and, more particularly, to a binary state machine system and method for REGEX processing of a data stream in an intrusion detection system.

BACKGROUND OF THE INVENTION

Security is a major concern among operators of computer networks. Attacks upon computer networks can disrupt the service of the computer network and can potentially expose proprietary data to such persons committing attacks.

Network security products such as intrusion detection systems (ID systems) can use a passive filtering technique to detect policy violations and/or patterns of misuse that indicate an attack is occurring or is imminent. The passive filtering technique usually comprises monitoring traffic upon or outside of the computer network for packets of data and comparing these packets of data with known "attack signatures."

Some conventional ID systems use a string matching algorithm as a passive filtering technique. A string-matching algorithm takes a string of characters from the input stream and compares the string of characters to known attack signatures. For example, some conventional ID systems can use a UNIX-based regular expression (REGEX) engine to search through an input stream for character strings that match an attack signature. In such a system, first the input stream is buffered. The buffer is compared to the REGEX command which searches through the entire buffer in order to find a match. If a match is found, indicating an attack, an alarm is sounded or preventative measures are taken.

There are several disadvantages with such a conventional ID system. For example, excessive memory and CPU resources are required during buffer manipulation. Furthermore, it may be necessary to

examine individual characters in the data buffer several times in order to find a valid match. Furthermore, as data traffic increases such problems lead to the possibility of dropping packets, resulting in a potential failure to detect an attack.

An additional complicating factor is that for some attacks, there can be a large amount of irrelevant data between strings of relevant data. As result, these conventional systems can require the buffering of large portions of the input stream, and it can be necessary to search portions these buffered portions multiple times. Furthermore, it can be possible for an attack to cross a buffer boundary, leading to the possibility of missing the attack.

SUMMARY OF THE INVENTION

In accordance with the present invention, a binary state machine system and method are disclosed that provide significant advantages over prior developed string searching algorithms for intrusion detection systems.

According to one aspect of the present invention, a method for using a binary state machine for processing a data stream in an intrusion detection system comprises maintaining a state table. The state table is indexed such that inputs comprising a current state and a current character yield an output of a new state. The new state is related to an indication of an attack on a computer network. The method further includes maintaing a current state. An input stream comprising a plurality of characters is received. A first character of the input stream is selected as the current character. The current character and the current state are compared to the state table to generate a new state.

According to another aspect of the present invention, a system for use as a binary state machine for processing a data stream in an intrusion detection system comprises a state table. The state table is indexed such that inputs comprising a current state and a current character yield an output of a new state. The new state is related to an attack on a computer network. The system further comprises a state machine communicatively coupled to the state table. The state machine is operable to maintain the current state and receive an input stream comprising a plurality of characters. The state machine is further operable to select a first

character of the input stream as the current character and compare the current character and the current state to the state table to generate a new state.

It is a technical advantage of the present invention that each character in the input stream need be examined only once. This advantage reduces the need for excessive buffering of data, and preserves CPU and memory resources of an ID system incorporating the invention.

It is another technical advantage that the present invention allows for more efficient searching of the input stream. This reduces the possibility of dropping packets, and increases the probability of detecting attacks.

It is a further technical advantage that it accounts for irrelevant data between strings of relevant data in an input stream without the need for excessive buffering of data. This eliminates the problem in conventional systems that occurs when an attack crosses the buffer boundary.

Other technical advantages should be apparent to one of ordinary skill in the art in view of the specification, claims, and drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

A more complete understanding of the present invention and advantages thereof may be acquired by referring to the following description taken in conjunction with the accompanying drawings, in which like reference numbers indicate like features, and wherein:

FIGURE 1 is a block diagram of a computer network including an intrusion detection system having one embodiment of a state machine for processing data streams;

FIGURE 2 is a flow diagram of one embodiment of a method for processing data streams using a binary state machine; and

FIGUREs 3A through 3C are block diagrams showing the operation of one embodiment of a binary state machine for an example data stream.

DETAILED DESCRIPTION OF THE INVENTION

FIGURE 1 is a block diagram of a computer network including an intrusion detection system having one embodiment of a state machine for processing data streams. A computer network, indicated generally at 10, includes a protected network 12 and an unprotected network 14. Protected network 12 is a "internal" network, meaning that only authorized users are permitted to pass data between systems coupled within protected network 12.

Protected network 12 includes several network devices coupled to a network medium 22. In the embodiment of FIGURE 1, the network devices include a workstation 20, a server 23, a router 25, and a printer 27. Network 22 can include any network medium and protocol thereon, such as Ethernet. Firewall 18 couples to network medium 22 and separates protected network 12 from an unprotected network 14. Firewall 18 is intended to operate as a gate keeper, preventing unauthorized users on unprotected network 14 to interfere with the operation of protected network 12. Firewall 18 further allows authorized users on unprotected network 14 to access protected network 12. In the example of FIGURE 1, unprotected network 14 includes network medium 28 coupled to the Internet 26.

An intrusion detection system 30 couples to network medium 22, network medium 28, or firewall 18. Intrusion detection system 30 may comprise, for example, software code executing on a computing platform separate from other network devices. Alternatively, intrusion detection system 30 can include functionality integrated

within firewall 18 or some other network device on protected network 12.

Intrusion detection system 30 has access to network traffic on computer medium 28 and/or computer medium 22. As shown, intrusion detection system 30 can couple to network medium 28 on unprotected network 14. Alternatively, as further shown in FIGURE 1, intrusion detection system 30 can couple to firewall 18, wherein firewall 18 passes network traffic to intrusion detection system.

Intrusion detection system 30 includes state machine 32 communicatively coupled to state tables 36. Intrusion detection system 30 and state machine 32 can comprise, for example, executable code stored on a computer readable medium and executable by a processor communicatively coupled to the computer readable medium. Alternatively, such components can be implemented in firmware or an ASIC implementation.

In operation, network traffic travels on network medium 28. The network traffic accessing firewall 18 and attempting to access firewall 18 form an input stream 29 to intrusion detection system 30. For example, input stream 29 may include data intended, for example, to one of the network devices within protected network 12. However, input stream 29 may also include some type of attack upon protected network 12.

Intrusion detection system 30 can recognize if input stream 29 includes an attack signature, meaning data within input stream 29 indicates an attack is occurring on protected network 12. Intrusion detection system

detects the hostile nature of data within input stream 29 by practicing the present invention.

Ins a2

Intrusion detection system 30 access input stream 29 and communicates input stream 29 to state machine 32. State machine 32 maintains a current state 33. State machine 32 further selects a character 38 from input stream 29 as a current character. State machine 32 then compares current state 33 and current character 38 to state tables 36 to determine a new state 35.

State tables 36 are indexed such that inputs comprising the current state and a current character yield an output of a new state. State tables 36 are formed such that the new state can be related to an indication of an attack on a computer network, as explained with respect to later FIGUREs.

By using such an implementation, intrusion detection system 30 improves the efficiency of detection of attack signatures over conventional intrusion detection systems. For example, each character in the input stream is processed by intrusion detection system 30 only once. After the new state based upon that character and the current state is determined, there is no longer a need to buffer that character. As such, the implementation of FIGURE 1 eliminates the drain upon processor and memory resources taken up by buffer manipulation.

FIGURE 2 is a flow diagram of one embodiment of a method for processing data streams using a binary state machine. The method starts at step 50. At step 54, a REGEX attack signature is defined. At step 58, a state table is created corresponding to that REGEX attack signature. The objective of step 58 is to create a state

table indexed such that if a current state is known and a current character is known a new state can be generated. The REGEX command can include, for example, a character string or combination of character strings that indicate a particular attack is occurring upon a computer network. There are many such strings and many such attacks known in the art.

At step 62, the system is initialized and the "current state" is set. Such a step may be necessary, for example, before the input stream is detected. At step 66, the data stream is received. At step 68, the "current character" is selected as a character of the data stream received in step 66. For example, the first such character in the data stream can be selected as the first "current character."

At step 70, the "current state" and "current character" of the data stream are compared to the state table in order to generate a "new state." At step 74, the system checks to see if the "new state" is equal to an state indicating an attack is occurring. Most, if not all, attacks will have a signature comprise more than one character. Therefore, the state table to detect such an attack will include more than one state. If an attack is detected, at step 78 an alarm is generated or a response is created. For example, such an alarm may be an indication transmitted to an operator on the network. A response could include the implementation of a countermeasure--for example resetting a connection. If the new state does indicate an attack at step 74, the method continues to step 82.

At step 82, it is decided whether or not to continue the method. If not, the process ends at step 90. If the method continues, at step 86 the "current state" is set to equal the "new state." Then, at step 88, the "current character" is set to equal the next character in the data stream. The method then continues to step 70 to repeat the comparing step.

As can be seen in the method of FIGURE 2, each character in the data stream is input into the state table along with the current state to create a new state. This new state is then used with the next character in the data stream to create another new state.

The method further shows how an intrusion detection system implementing such a system can attain many advantages. The need for extensive data buffering is eliminated, because each character need only be examined a once, and compared to the state table once. Such a system implementing the method of FIGURE 2 would improve efficiency as it would require fewer processing and memory resources. As such, an intrusion detection system employing the method of FIGURE 2 will have fewer instances of dropped packets or missed signatures as compared to conventional intrusion detection systems.

FIGUREs 3A through 3C are block diagrams showing the operation of one embodiment of a binary state machine for an example data stream.

FIGURE 3A shows an example of an input stream 100. In the example of FIGURE 3A, input stream 100 includes the characters: "the dog ran home." FIGURE 3A further shows the REGEX command 104 of an attack signature. In the example of FIGURE 3A, the REGEX command is

"dog.*home." This REGEX command indicates that an attack is indicated if: (1) the characters "d", "o", "g" are observed in order; (2) any number of characters occur after the "g"; and then (3) the characters "h", "o", "m", and "e" occur in order. As can be seen, with this attack signature, a properly executing intrusion detection system will recognize that input stream 100 indicates an attack is occurring on the computer network.

FIGURE 3B shows one embodiment of a state table formed from the REGEX command 104 of FIGURE 3A. State table 112 includes a state index 129 and a character index 120. State index 129 corresponds to the "current state" as referenced in the method of FIGURE 2. Character index 120 corresponds to the "current character" as referenced in the method of FIGURE 2. In the embodiment of FIGURE 2, character index 120 comprises ASCII codes.

State table 112 is indexed such that inputs comprising a current state and a current character yield an output of a new state. The new state is related to an indication of an attack on a computer network. In the embodiment of FIGURE 2, state number 8 is the state at which the "dog.*home" attack signature is detected.

In operation, given a current state 129 and a current character 120, table 112 generates a new state. For example, if the current state 124 is "1" and the current character 120 is "O", represented by ASCII code "79," the new state is "2." The state of "2" does not yet indicate an attack is occurring. Next, the current state 124 is set to "2" and the current character 120 is set to the next character in the data stream. As seen by

A5 cont

state table 112, if the next character in the data stream is any character except for "G", the new state will be "0." This indicates that the attack this state table is designed to detect requires a "G" to immediately succeed an "O." On the other hand, if the next character after the "O" is a "G", it can be seen that state table 112 will generate a new state of "3."

FIGURE 3C shows the operation of the state machine of the current invention using the input stream 100 of FIGURE 3A, the REGEX command 104 of FIGURE 3A, and the state table 112 of FIGURE 3B. For example, in the first block, current state 125 is "0", and current character 127 "T", which is the first character in data stream 100. A new state 129 is generated by these inputs indexing state table 112. New state 129 with these inputs is "0". FIGURE 3C shows the current state 125, current character 127, and new state 129 of each character of data stream 100 in FIGURE 3A.

The present invention is described with respect to ASCII character sets. However, the present invention contemplates that an intrusion detection system implementing the method of the invention can be capable of interpreting character sets other than the ASCII character sets. Additionally, the present invention contemplates that many state tables can be integrated into a single intrusion detection system, such that many different attack signatures can be detected.

Although the present invention has been described in detail, it should be understood that various changes, substitutions and alterations can be made thereto without

departing from the spirit and scope of the invention as defined by the appended claims.